Back to Reproducibility in 3D Anomaly Detection: Revisiting RGB-D for Point-Cloud Evaluation

Remi Lhoste[1,2][0009−0006−4172−0460], Damien Delhay[2], and Antoine Vacavant[1][0000−0001−9616−3282]

[1] Institut Pascal, Universite Clermont Auvergne, UMR 6602 UCA/SIGMA/CNRS, 63171 Aubiere, France
antoine.vacavant@uca.fr
[2] O2game, 60200 Compiegne, France
{remi.lhoste, damien.delhay}@o2game.com

Abstract. Reproducibility in 3D anomaly detection is complicated by the reuse of baselines across sensing representations. Several widely used methods were originally introduced on organized RGB-D benchmarks, where important stages of the pipeline are implicitly defined by the image grid. When these baselines are later reported on point-cloud-only benchmarks, the transfer is not unique: subsampling, neighborhood construction, aggregation, patch formation, and localization must be redefined explicitly. As a result, the same baseline name may refer to operationally different pipelines, weakening scientific comparability across papers. We study this reproducibility issue through the `BTF/FPFH` baseline. We identify the representation-dependent components of the original RGB-D implementation, formalize a fully specified point-cloud transfer protocol, and release an open reference implementation designed for reuse across datasets. We first evaluate the transfer in a controlled setting where organized RGB-D and transferred point-cloud executions can be compared on the same samples and masks. We then reuse the same protocol on point-cloud-only benchmark settings and compare it with previously reported point-cloud adaptations. Our results show that transfer choices materially affect the apparent strength of the baseline, even when its descriptor family and nearest-neighbor scoring rule are preserved. Beyond the specific case of `BTF/FPFH`, this work argues that cross-representation baseline reuse requires not only open code, but also an explicit operational protocol. The contribution is therefore a reproducible artifact and reference baseline that supports fairer evaluation on future point-cloud anomaly detection benchmarks. (code is available at `github.com/Rlhoste/3D-ADS`)

Keywords: 3D anomaly detection · reproducibility · point clouds

1 Introduction

MVTec 3D-AD [1] made organized RGB-D data a central evaluation setting for 3D anomaly detection. In this setting, depth is available on an image grid, localization masks are image-aligned, and several stages of a baseline pipeline are implicitly defined by this regular structure. As the field has expanded toward point-cloud-only benchmarks such as Real3D-AD [7] and Anomaly-ShapeNet [6], many RGB-D baselines have been reused under a different sensing representation. This transfer creates a reproducibility issue: the problem is not only whether the baseline code exists, but whether the transferred baseline is still defined unambiguously across representations.

This ambiguity arises because an RGB-D baseline does not transfer to point clouds by changing only the input loader. Operations that were previously fixed by the image grid, including subsampling, neighborhood construction, local aggregation, patch formation, and localization, must now be redefined explicitly. As a result, the same baseline name may correspond in practice to different operational pipelines, weakening scientific comparability across papers and making later benchmark conclusions harder to interpret.

This paper studies that issue through the `BTF/FPFH` branch [5,11]. We identify which components of the original RGB-D baseline depend on the organized representation, and we replace them with a fully specified point-cloud transfer protocol while preserving the nearest-neighbor scoring logic of the method. We then evaluate this protocol in two complementary settings: first on MVTec 3D-AD, where the original RGB-D execution and the transferred point-cloud execution can be compared on the same samples and masks, and then on Real3D-AD and Anomaly-ShapeNet, where we compare the resulting reference baseline with previously reported point-cloud adaptations.

Our goal is therefore not only to publish another transferred implementation, but to make the transfer itself reproducible, inspectable, and reusable as a scientific reference for future point-cloud anomaly detection benchmarks.

Our contributions are:

- We identify the grid-dependent components of the original `BTF/FPFH` branch.
- We propose a point-cloud transfer protocol that replaces these components explicitly.
- We validate the transfer on MVTec 3D-AD in a controlled RGB-D/point-cloud setting.
- We report reference results on Real3D-AD and Anomaly-ShapeNet.

2 The Original `BTF/FPFH` RGB-D Baseline

`BTF` (Back to the Feature) was introduced on MVTec 3D-AD as a study of feature representations for 3D anomaly detection [5]. The original work compares several ways of representing RGB-D samples, including RGB and depth ImageNet features [3], raw geometric values, HOG [2], Dense SIFT [8], FPFH [11],

and combinations of color and geometric descriptors. Its main result is that handcrafted geometric descriptors, especially FPFH [11], provide a strong representation for geometric anomaly detection, and that combining FPFH with color features yields the full `BTF` model.

This paper focuses on the `FPFH` branch rather than the full color-plus-geometry model. This branch is important because it is training-free, purely geometric, and has been reused as a baseline in point-cloud anomaly detection benchmarks. It therefore provides a useful case study for reproducibility: the anomaly score is based on a simple nearest-neighbor comparison of local descriptors, but the way these descriptors are constructed depends on the representation used by the implementation.

In the released RGB-D implementation, the `FPFH` branch is still organized around the MVTec 3D-AD image grid. The input point cloud is stored as an organized depth-derived grid aligned with the RGB image and localization mask. FPFH descriptors are computed from valid 3D points, but the descriptor values are written back into an organized feature map. Local smoothing, patch construction, and anomaly-map generation are then performed with image-grid operations inherited from the PatchCore-style pipeline [10]: feature maps are averaged locally, pooled into patch descriptors, compared to nominal training patches, and projected back to an image-aligned score map.

This design is natural for MVTec 3D-AD, where the data and metrics are image-aligned. However, it also means that the `BTF/FPFH` branch is not defined by the FPFH descriptor and nearest-neighbor score alone. Subsampling, local aggregation, patch formation, and localization output are all partly defined by the organized RGB-D grid. When the same baseline is reused on point-cloud-only benchmarks, these grid-dependent stages must be replaced explicitly rather than assumed to transfer automatically.

3 Fully Specified Point-Cloud Transfer Protocol

Figure 1 summarizes the transfer protocol defined in this paper. Starting from the RGB-D `BTF/FPFH` branch described above, we keep the descriptor family, memory-bank construction, and nearest-neighbor anomaly scoring unchanged. We replace only the stages whose definition depended on the organized RGB-D grid: subsampling, descriptor aggregation, patch construction, and output projection.

3.1 Point-Cloud Subsampling

The RGB-D branch obtains its descriptors from an organized image grid. In the point-cloud branch, we extract all valid 3D points, normalize them to an object-scale coordinate frame, and apply voxel-grid subsampling. Normalization is part of the protocol because `FPFH` uses metric neighborhoods: the normal-estimation radius is set to `5%` of the object extent after normalization. On MVTec 3D-AD, we target `16384` points, matching the order of magnitude induced by

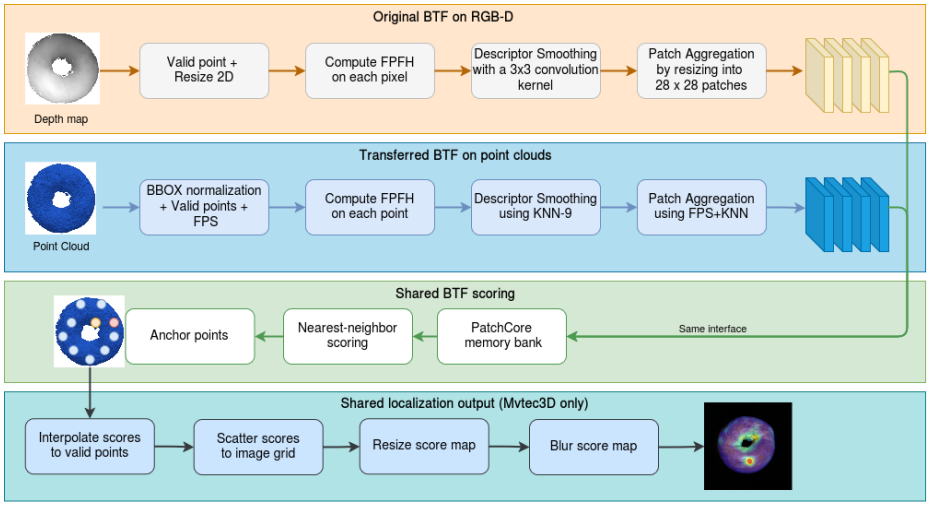

Fig. 1. Overview of the proposed representation transfer. The adapted branch replaces the representation-dependent subsampling, aggregation, patch construction, and output projection stages, while preserving the original `BTF/FPFH` nearest-neighbor scoring logic.

the original grid resize. The voxel size is calibrated once on the `bagel` class and reused across classes, fixing a spatial resolution rather than tuning each category independently.

3.2 Descriptor Aggregation

The RGB-D branch smooths descriptors with local image-grid averaging. We replace this operation by nearest-neighbor averaging in 3D space: after computing `FPFH` descriptors on the subsampled point cloud, each descriptor is averaged with its `k=9` nearest descriptor neighbors. The value `k=9` mirrors the original `3 x 3` image neighborhood while making the neighborhood geometric rather than image-based.

3.3 Patch Formation and Scoring

The RGB-D branch pools the organized feature map into a regular patch lattice before PatchCore-style scoring. In the point-cloud branch, we replace this lattice by `784` farthest-point-sampling anchors, matching the number of cells in the original `28 x 28` patch grid. Each anchor descriptor averages the `100` nearest smoothed descriptors around the anchor. The resulting descriptors are scored with the same PatchCore memory-bank and nearest-neighbor comparison as the original branch; only the construction of the descriptors being scored changes.

3.4 Localization Projection

The scoring step produces anomaly scores at the patch anchors. We project these scores back with KNN interpolation using `k=4`, a small local neighborhood analogous to bilinear projection on a grid. On MVTec 3D-AD, the interpolation targets are the valid points of the downsampled image-grid point cloud; the interpolated scores are scattered back to their image indices before the same resize and blur as the RGB-D branch, enabling direct pixel-level AUROC and AU-PRO comparison. On point-cloud-only benchmarks, the interpolation targets are the original valid input points, so localization is evaluated directly with point-level scores.

Overall, the protocol turns an implicit adaptation into a reproducible function: normalization, subsampling, descriptor smoothing, anchor construction, patch neighborhoods, scoring, and output projection are all explicit method-defining choices.

4 Experimental Protocol

4.1 Datasets and Metrics

We evaluate the transfer protocol in two complementary settings. MVTec 3D-AD [1] is used as the controlled representation-transfer setting, because it allows the point-cloud protocol to be compared against the original organized RGB-D execution of the same baseline. We also evaluate on point-cloud-only benchmark protocols, including Real3D-AD [7] and Anomaly-ShapeNet [6], to report the baseline in the setting where future point-cloud anomaly detection datasets are expected to use it.

On MVTec 3D-AD, we report image-level ROCAUC, pixel-level ROCAUC, AU-PRO, mean inference time per test sample in milliseconds, and mean peak VRAM per test sample in MiB. The first three validate anomaly detection behavior; the last two document computational cost. For Real3D-AD and Anomaly-ShapeNet, we follow the metrics used by the corresponding benchmark protocols so that the transferred baseline can be compared to reported point-cloud results.

4.2 Reproduction Path

The reference implementation is executed through an explicit method-selection interface. The original organized-grid baseline is run without point-cloud transfer:

```
python 3D-ADS/main.py --methods "FPFH"
```

For the point-cloud transfer, the public baseline command is:

```
python 3D-ADS/main.py --methods "FPFH" --point-cloud
```

The second command instantiates the transfer protocol described above. It evaluates the same `FPFH` branch on the point-cloud representation instead of silently starting unrelated RGB baselines. Optional command-line arguments expose the subsampled point count, patch count, neighborhood sizes, interpolation parameter, voxel calibration, and coreset policy used by the point-cloud branch rather than leaving them as hidden defaults. The evaluation loop also records elapsed inference time, peak GPU memory allocated during prediction, and the standard anomaly detection outputs used for image-level and localization metrics. These parameters define the reported accuracy values; acceleration backends may change runtime, but they do not define a different baseline.

For the point-cloud-only benchmarks, we use the same transferred branch but expose the dataset-specific point counts explicitly. Real3D-AD is evaluated with:

```
python 3D-ADS/main.py --dataset real3dad --methods FPFH \
  --point-cloud --npoints 150000 \
  --num-patches 4000 \
  --class-voxel --coreset 1
```

Anomaly-ShapeNet is evaluated with:

```
python 3D-ADS/main.py --dataset anomalyshapenet --methods FPFH \
  --point-cloud --class-voxel \
  --npoints 150000 --num-patches 4000 \
  --coreset 1
```

5 Validation Results

5.1 Controlled Transfer on MVTec 3D-AD

MVTec 3D-AD is used as a controlled transfer setting. Both executions are evaluated on the same test samples and ground-truth masks, and the point-cloud scores are projected back to the same image grid used for the benchmark metrics. The comparison therefore isolates the effect of changing the internal representation while keeping the final evaluation protocol fixed. Fig. 2 reports image ROCAUC, pixel ROCAUC, and AU-PRO for each category; the full numerical table is given in Table 3 in the Appendix.

The transferred point-cloud execution obtains a higher mean image ROCAUC than the organized RGB-D execution (`0.835` vs. `0.779`), while pixel ROCAUC and AU-PRO remain close (`0.973` vs. `0.978`, and `0.916` vs. `0.924`). These results show that the transfer keeps the baseline comparable, but the class-wise gaps in Fig. 2 also show that the conversion is not neutral. The same PatchCore nearest-neighbor scoring rule is applied, but the descriptors scored by PatchCore are built differently: the RGB-D branch relies on grid-based feature smoothing and adaptive pooling, whereas the point-cloud branch uses normalized 3D neighborhoods, KNN feature smoothing, FPS anchor selection, KNN

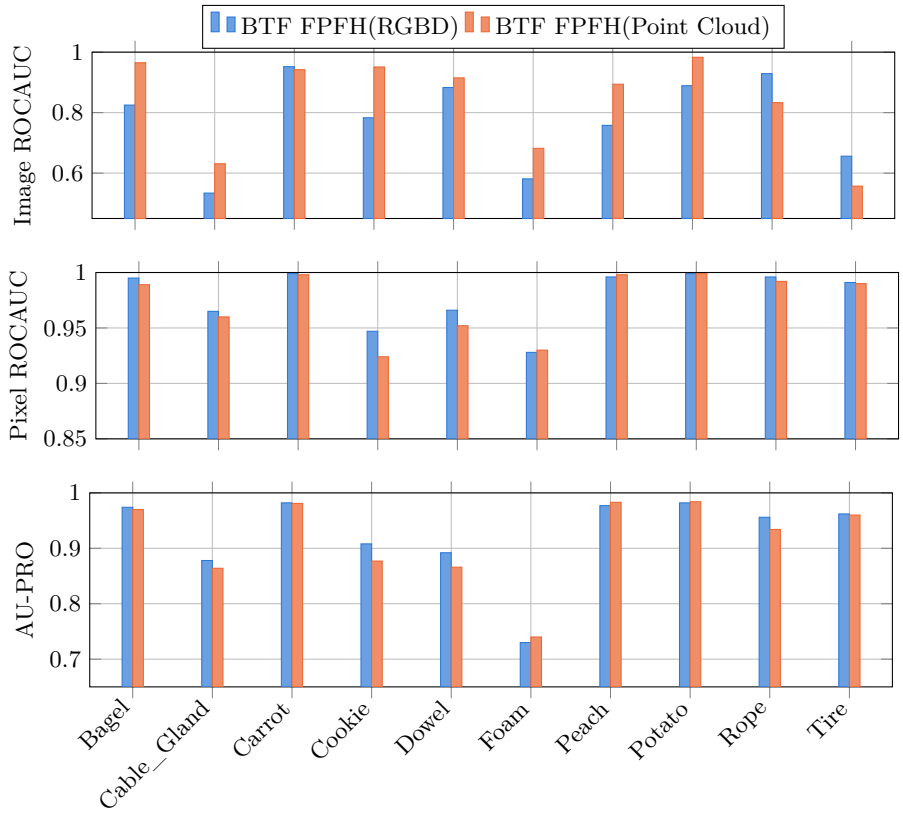

Fig. 2. Controlled transfer on MVTec 3D-AD. Both executions use the same samples and masks; point-cloud scores are projected back to the benchmark image grid. Grouped bars report the class-wise changes induced by replacing image-grid operations with explicit 3D neighborhood operations.

patch descriptor construction, and KNN interpolation of anchor scores back to the evaluation grid.

Fig. 3 illustrates the practical effect of this change. In the RGB-D branch, local neighborhoods follow the regular image lattice: points that are close in the projected image contribute to nearby descriptors and patches. In the point-cloud branch, neighborhoods are defined by 3D Euclidean distance. Points that appear close after projection may be far apart on the object surface, while points that are close in 3D may not form the same neighborhood in the image grid. Shallow dents, boundaries, and oblique surfaces can therefore produce sharper, weaker, or spatially shifted anomaly responses after transfer. This controlled experiment highlights the reproducibility issue addressed by the reference baseline: for point-cloud baselines, the descriptor and scoring rule are not sufficient to define the

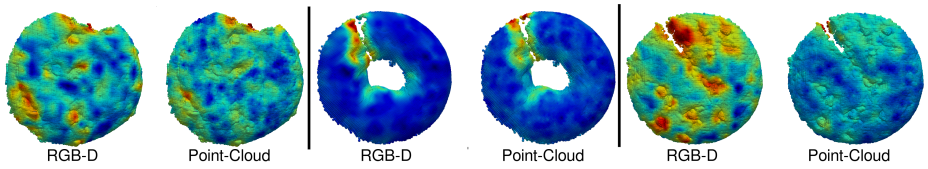

Fig. 3. Qualitative localization differences after transfer. Replacing image-grid neighborhoods with 3D k-nearest-neighbor operations changes anomaly maps even when the descriptor family and PatchCore scoring rule are preserved.

method; the construction of scored descriptors and their projection back to the evaluation domain must also be specified.

Runtime and memory are also part of the controlled transfer study. Table 1 reports MVTec 3D-AD inference costs for the organized RGB-D branch and for the transferred point-cloud branch. The comparison is not only an implementation detail: replacing grid operators with point-cloud processing changes the computational primitives. RGB-D patch construction is mostly regular pooling on a fixed image lattice, while the point-cloud branch requires voxel subsampling, 3D neighborhood queries, and farthest-point anchor selection.

The public reference implementation keeps the method definition fixed while relying on released PyTorch/Open3D components [9,12]. Table 1 reports four execution modes forming an implementation gradient: RGB-D, point-cloud transfer on CPU (`PC_CPU`), the same transfer using GPU tensor operations without custom kernels (`PC_GPU`), and an internal acceleration reference using custom CUDA kernels for FPFH, KD-tree KNN queries, and bucket-pruned farthest-point sampling [4] (`PC_Cuda_ops`). Subsample still uses the same Open3D CPU voxel-downsampling call in every mode. The main result is that `PC_Cuda_ops` matches the original RGB-D branch's peak VRAM (`115.7` vs. `97.0` MiB) while remaining about `2x` slower overall (`110.4` vs. `50.1` ms): optimized CUDA kernels narrow but do not remove the representation-transfer overhead, since point-cloud neighborhood search and sampling stay structurally costlier than grid pooling. `PC_CPU` and `PC_GPU` are part of the public reference implementation; `PC_Cuda_ops` is not, and should be read only as an acceleration reference: it changes runtime and memory, not the baseline definition or the transfer protocol.

Table 1. Mean per-stage inference time (ms) of BTF FPFH on MVTec 3D-AD, averaged over all classes, plus total time, peak VRAM, and peak host RAM. `PC_CPU` and `PC_GPU` are public execution modes of the point-cloud transfer; `PC_Cuda_ops` is an internal acceleration reference, not the default public reference implementation. Stage times do not sum exactly to the total due to unmeasured glue code and rounding.

Stage	RGB-D	PC_CPU	PC_GPU	PC_Cuda_ops
Valid Point Extraction	0.2	1.3	1.2	1.2
Subsample	1.2	29.5	30.0	28.9
Descriptor Computation	28.9	52.6	51.1	7.0
Descriptor Smoothing	1.9	518.8	11.9	0.8
Patch Construction	0.2	112.3	27.1	30.3
Scoring	9.3	25.1	34.6	34.2
Localization Projection	1.0	15.9	1.6	1.4
Total (ms)	50.1	791.4	164.1	110.4
Peak VRAM (MiB)	97.0	0.0	763.8	115.7
Peak RAM (MiB)	2034.5	4230.2	2881.5	2231.9

5.2 Reproducibility Gap on Point-Cloud-Only Benchmarks

We next apply the specified point-cloud branch to point-cloud-only benchmark protocols. This setting differs from MVTec 3D-AD: there is no paired organized-grid execution on the same samples, so the comparison cannot isolate representation transfer under a fixed evaluation grid. Instead, it tests whether previously reported point-cloud `BTF/FPFH` baselines correspond to the same operational method once the RGB-D pipeline is ported carefully to point clouds.

This distinction matters because our code inspection indicates that existing point-cloud benchmark baselines using the `BTF/FPFH` name implement simplified point-cloud variants rather than the full representation transfer studied here. The Real3D-AD `FPFH` implementation centers each point cloud but does not normalize it to a common object extent, computes radius-based descriptors in dataset coordinates, samples descriptors by point-index stride, and scores sparse point descriptors directly. Anomaly-ShapeNet follows this Real3D-AD-style baseline protocol. As a result, a published `BTF/FPFH` number may reflect an underspecified point-cloud variant rather than a controlled transfer of the original RGB-D method.

Table 2. Point-cloud-only benchmark reproducibility gap. Published columns report `BTF/FPFH` means from existing point-cloud benchmark tables; ours reports the fully specified point-cloud transfer used in this paper. I-AUC denotes image-level AUROC and P-AUC denotes point-level AUROC.

Dataset	Pub. I-AUC	Ours I-AUC	Pub. P-AUC	Ours P-AUC
Real3D-AD	0.603	0.688	0.566	0.865
Anomaly-ShapeNet	0.528	0.880	0.628	0.912

The gap in Table 2 shows why this distinction is important. When the point-cloud version is fully specified, including choices such as scale normalization, subsampling, local descriptor smoothing, patch formation, coreset selection, and score projection, the resulting baseline can differ substantially from published point-cloud variants that share the same high-level name. This comparison should not be read as a claim that previous benchmark results are invalid; it shows that the same baseline name can denote operationally different pipelines unless the transfer protocol is specified. The descriptor family is unchanged, but the evaluated pipeline is not. The consequence is methodological: underspecified transfers can make a baseline appear weaker than it is, leading to misleading conclusions about the strength of later methods. The contribution of this work is therefore to provide a clearer and reusable way to port RGB-D baselines to point-cloud-only benchmarks while preserving the methodological structure needed for fair comparison.

6 Discussion and Conclusion

This work is motivated by a reproducibility issue that goes beyond the specific case of `BTF/FPFH`. In 3D anomaly detection, several baselines were first established on organized RGB-D data, where important stages of the pipeline are implicitly defined by the image grid. When these baselines are reused on point-cloud-only benchmarks, the transfer is not uniquely determined. As a result, the same baseline name may correspond to different operational pipelines, weakening scientific comparability across papers.

Using the `BTF/FPFH` branch as a case study, we identified the representation-dependent stages of the original baseline, formalized a fully specified point-cloud transfer protocol, and released a reusable reference implementation. Our experiments show that transfer choices can materially affect the apparent strength of the baseline, even when the descriptor family and nearest-neighbor scoring rule are preserved.

The contribution of this work is therefore not only an implementation, but a reproducible operational definition of a widely used baseline across representations. More broadly, this case study shows that when methods cross representation boundaries, open code alone is not sufficient: the transferred operations, their parameters, and their evaluation projection must also be specified explicitly if a baseline is to remain a meaningful scientific reference.

References

1. Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C.: The MVTec 3d-AD dataset for unsupervised 3d anomaly detection and localization. In: Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Volume 5: VISAPP. pp. 202–213 (2022)

2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). pp. 886–893 (2005)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 248–255 (2009)
4. Han, M., Wang, L., Xiao, L., Zhang, H., Zhang, C., Xu, X., Zhu, J.: QuickFPS: Architecture and algorithm co-design for farthest point sampling in large-scale point clouds. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **42**(11), 4011–4024 (2023)
5. Horwitz, E., Hoshen, Y.: Back to the feature: Classical 3d features are (almost) all you need for 3d anomaly detection. arXiv preprint arXiv:2203.05550 (2022). https://doi.org/10.48550/arXiv.2203.05550
6. Li, W., Xu, X., Gu, Y., Zheng, B., Gao, S., Wu, Y.: Towards scalable 3d anomaly detection and localization: A benchmark via 3d anomaly synthesis and a self-supervised learning network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22207–22216 (2024)
7. Liu, J., Xie, G., Chen, R., Li, X., Wang, J., Liu, Y., Wang, C., Zheng, F.: Real3D-AD: A dataset of point cloud anomaly detection. In: Proceedings of the Thirty-seventh Conference on Neural Information Processing Systems, Datasets and Benchmarks Track (2023)
8. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)
9. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32 (NeurIPS). pp. 8024–8035 (2019)
10. Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14318–14328 (2022)
11. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3d registration. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp. 3212–3217 (2009)
12. Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3d data processing. arXiv:1801.09847 (2018)

A Full Per-Class Results

Table 3 reports the numeric per-class values underlying Fig. 2. Table 4 reports Real3D-AD mean comparisons and per-class scores. Table 5 provides detailed Anomaly-ShapeNet comparisons so that future work can cite or reuse the same reference values.

Table 3. MVTec 3D-AD image-level and localization scores, measured with our reference implementation. I denotes image-level ROCAUC, P pixel-level ROCAUC, and PRO the AU-PRO metric.

Class	RGB-D I	Ours I	RGB-D P	Ours P	RGB-D PRO	Ours PRO
Bagel	0.825	0.965	0.995	0.989	0.974	0.970
Cable_Gland	0.534	0.631	0.965	0.960	0.878	0.864
Carrot	0.952	0.942	0.999	0.998	0.982	0.981
Cookie	0.783	0.951	0.947	0.924	0.908	0.877
Dowel	0.883	0.915	0.966	0.952	0.892	0.866
Foam	0.581	0.682	0.928	0.930	0.730	0.740
Peach	0.758	0.894	0.996	0.998	0.977	0.983
Potato	0.889	0.983	0.999	0.999	0.982	0.984
Rope	0.929	0.833	0.996	0.992	0.956	0.934
Tire	0.656	0.557	0.991	0.990	0.962	0.960
Mean	0.779	0.835	0.978	0.973	0.924	0.916

Table 4. Detailed Real3D-AD scores. Published columns report `BTF`/`FPFH` values from benchmark tables; ours are measured with the fully specified point-cloud transfer. I-AUC denotes image-level AUROC and P-AUC point-level AUROC.

Class	Pub. I	Ours I	Pub. P	Ours P	Class	Pub. I	Ours I	Pub. P	Ours P
Airplane	0.520	0.591	0.564	0.826	Fish	0.549	0.580	0.514	0.880
Candybar	0.630	0.905	0.735	0.966	Gemstone	0.648	0.898	0.597	0.979
Car	0.560	0.530	0.647	0.896	Seahorse	0.779	0.431	0.520	0.637
Chicken	0.432	0.673	0.608	0.792	Shell	0.754	0.539	0.489	0.808
Diamond	0.545	0.994	0.563	0.978	Starfish	0.575	0.854	0.392	0.721
Duck	0.784	0.557	0.601	0.961	Toffees	0.462	0.709	0.623	0.934
Mean	0.603	0.688	0.566	0.865					

Table 5. Detailed Anomaly-ShapeNet scores. Published `BTF(FPFH)` values are included as references; ours are measured with the fully specified point-cloud transfer. I-AUC denotes image-level AUROC and P-AUC point-level AUROC.

Class	Pub. I	Ours I	Pub. P	Ours P	Class	Pub. I	Ours I	Pub. P	Ours P
Ashtray0	0.420	1.000	0.596	0.900	Headset0	0.520	0.982	0.620	0.912
Bag0	0.546	0.957	0.746	0.980	Headset1	0.490	1.000	0.591	0.977
Bottle0	0.344	1.000	0.641	0.987	Helmet0	0.571	0.658	0.575	0.923
Bottle1	0.546	1.000	0.549	0.779	Helmet1	0.719	0.976	0.749	0.871
Bottle3	0.322	0.968	0.622	0.868	Helmet2	0.542	0.707	0.643	0.909
Bowl0	0.509	1.000	0.710	0.992	Helmet3	0.444	0.885	0.724	0.966
Bowl1	0.668	0.833	0.620	0.845	Jar0	0.424	1.000	0.427	0.996
Bowl2	0.510	0.919	0.518	0.908	Microphone0	0.671	1.000	0.675	0.981
Bowl3	0.490	0.904	0.690	0.988	Shelf0	0.609	0.649	0.619	0.878
Bowl4	0.609	0.781	0.679	0.831	Tap0	0.560	0.682	0.568	0.764
Bowl5	0.699	0.768	0.699	0.945	Tap1	0.546	0.593	0.641	0.777
Bucket0	0.401	0.975	0.401	0.753	Vase0	0.342	0.963	0.642	0.908
Bucket1	0.633	0.822	0.633	0.889	Vase1	0.219	0.824	0.619	0.719
Cap0	0.618	0.604	0.730	0.926	Vase2	0.546	1.000	0.646	0.996
Cap3	0.522	0.895	0.658	0.949	Vase3	0.699	0.739	0.699	0.847
Cap4	0.520	0.909	0.524	0.957	Vase4	0.510	0.918	0.710	0.961
Cap5	0.586	0.860	0.586	0.960	Vase5	0.409	0.957	0.429	0.931
Cup0	0.586	0.962	0.790	0.988	Vase7	0.518	0.924	0.540	0.975
Cup1	0.610	0.957	0.619	0.908	Vase8	0.668	0.906	0.662	0.938
Eraser0	0.719	1.000	0.719	0.979	Vase9	0.268	0.706	0.568	0.907
Mean	0.528	0.880	0.628	0.912					